**Front-end**

1. DisplayResult.java

This class takes the file-name as an argument and creates an instance of the class Computation and calls its methods.

1. Computation.java

This class recognizes three files using file-name as filename.py, filename\_passfail.txt and filename\_coverage.txt. The explanation of its methods are as follows.

1. codeToString()

This method reads the filename.py file line by line and stores it in a String array code. While adding each line to code[], a statement number (starting from 1) is appended at the beginning of each line and stored in code[] such that every line can be indexed in code[] using its statement number.

e.g. Statement 5 is stored in code[5].

1. getCount()

This method simply counts the number of lines in filename.passfail.txt file.

1. pfArray()

A passfail array is declared with length equal to the number of lines in filename\_passfail.txt file. For every line in the filename\_passfail.txt file, the first numeric value (run number) is stored in a variable ‘index’ and the other numeric value (0 or 1, which represents fail or pass) is stored in passfail[] at the location stored in index.

e.g. If run number 2 has passed, passfail[2]=1.

1. pfCal()

This method calculates the total number of runs that passed and the total number of runs that failed. It runs a for loop through each element of passfail[] and increments ‘pass’ if the value stored is 1 and increments ‘fail’ if the value is 0.

1. statementPassFail()

This method calculates the number of times each statement of the code passed as well as the number of times it failed.

It first creates an array temp\_array of size equal to the number of lines in filename\_coverage.txt file. A two dimensional array statement\_passfail is created to store the number of passes and fails for each statement. Each line in the filename\_coverage.txt file is read and checks if that same line exists in the temp\_array[]. If it exists, it does nothing and reads the next line. If it does not exist, the line is added in the next available location in temp\_array[]. Using space is delimiter the first and second numeric values in the line are stored in the variables statement\_no and run\_number respectively. To check whether this run\_number had passed or failed, it checks the value of passfail[run\_number] which would be 1 or 0 respectively. If it is 1, statement\_passfail[statement\_no][1] is incremented by 1 otherwise statement\_passfail[statement\_no][0] is incremented by 1.

1. getColorValue()

This method calculates the color value of each statement of the code ranging between 0 and 120.

For each statement, its passing percentage (passed) is calculated as number of runs in which it passes divided by the total number of passing runs. Its failing percentage (pfailed) is calculated similarly. These values are stored in a results array. The color values is calculated as (passed/(passed+pfailed))\*120.

1. createOutput()

The create() method of CreateHTML class is called.

1. CreateHTML.java

An html file is created with the name filename\_report.html.

1. RangeColor.java

(Mark)

1. Filename\_report.html

This file includes displays all the entire code with each statement highlighted by a color corresponding to the value calculated by the above formula. Comments are highlighted with gray color. Statements that are not executed by any run are not highlighted by any color. There is also a slide bar locked at the top of the page which can be used to change the value of a threshold such that the statements having their color value more than the threshold will be colored green and the remaining statements will be recolored based on a new scale ranging from 0 to the threshold value.